initial

Reads input files broadcasts, and allocates/initializes global variables.

[called by: focus, globals.] [calls: .]

Input namelist

The *focusin* namelist is the only input namelist needed for FOCUS running. It should be written to the file *example.input*, where 'example' is the argument passed by command line. Here are the details for the variables.

- `IsQuiet = -1`
  Information displayed to the user
  -2: more details & update unconstrained cost functions;
  -1: more details;
  0: essential;
  1: concise.
- `IsSymmetric = 0`
  Enforce stellarator symmetry or not
  0: no stellarator symmetry enforced;
  1: plasma periodicty enforced;
  2: coil and plasma periodicity enforced.

- `input_surf = 'plasma.boundary'`
  Input file containing plasma boundary information.
- `input_coils = 'none'`
  Input file containing initial guess for coils (in either format). If it is 'none' by default, it will be updated to 'coils.example' (case_init=-1) or 'example.focus' (case_init=0).
- `input_harm = 'target.harmonics'`
  Input file containing the target harmonics for Bmn optimization.

- `case_surface = 0`
  Specify the input plasma boundary format
  0: general VMEC-like format (Rbc, Rbs, Zbc, Zbs), seen in rdsurf;
  1: read axis for knots, seen in rdknot; (not ready)
- `knotsurf = 0.2`
  Minor plasma radius for knototrans, only valid for `case_surface = 1`
- `ellipticity = 0.0`
  Ellipticity of plasma for knototrans, only valid for `case_surface = 1`
- `Nteta = 64`
  Poloidal resolution for discretizing the plasma
- `Nzeta = 64`
  Toroidal resolution for discretizing the plasma

- `case_init = 0`
  Specify the initializing method for coils, seen in rdcoils
  -1: read the standard MAKEGRID format coils from `input_coils`;
  0: read FOCUS format data from `input_coils`;
  1: toroidally spaced `Ncoils` circular coils with radius of `init_radius`;
  2: toroidally spaced `Ncoils`-1 magnetic dipoles pointing poloidallly on the toroidal surface with radius of `init_radius` and a central infinitely long current. Dipole magnetizations anc the central current are all set to `init_current`.
- `case_coils = 1`
  Specify representation used for the initial coils, seen in rdcoils
  0: using piecewise linear representation; (not ready)
  1: using Fourier series representation;
- `Ncoils = 0`
  Number of coils initilized, only valid for `case_init = 1`
- `init_current = 1.0E6`
  Initial coil current (A), only valid for `case_init = 1`

- `init_radius = 1.0`
  Initial coil radius (m), only valid for `case_init = 1`
- `IsVaryCurrent = 1`
  Keep coil currents fixed or not, overriden by example.focus
  0: coil currents are fixed;
  1: coil currents are free;
- `IsVaryGeometry = 1`
  Keep coil geometries fixed or not, overriden by example.focus
  0: coil geometries are fixed;
  1: coil geometries are free;
- `NFcoil = 4`
  Number of Fourier coefficients for coils, only valid for `case_coils = 1`*, overriden by* example.focus
- `Nseg = 128`
  Number of segments for discritizing coils, only valid for `case_coils = 1`*, overriden by* example.focus

---

- `IsNormalize = 1`
  Normalizing coil parameters or not
  0: keep raw data (normalized to 1.0);
  1: currents being normalized to averaged absolute current, coil geometry parameters being normalized to major radius;
- `IsNormWeight = 1`
  each constraints normalized to initial value or not
  0: keep raw value for constraints;
  1: $w = w/f_0$ weights normalized to the initial values of each constraints;
- `case_bnormal = 0`
  Bn error normalized to $|B|$ *or not*
  0: keep raw Bn error;
  1: Bn residue normalized to local $|B|$;
- `case_length = 0`
  options for constructing coil length constraint, seen in length
  1: quadratic format, converging the target_length;
  2: exponential format, as short as possible;
- `weight_bnorm = 1.0`
  weight for Bn error, if zero, turned off; seen in bnormal
- `weight_bharm = 0.0`
  weight for Bn Fourier harmonics error, if zero, turned off; seen in bmnharm
- `weight_tflux = 0.0`
  weight for toroidal flux error, if zero, turned off; seen in torflux
- `target_tflux = 0.0`
  target value for the toroidal flux, if zero, automatically set to initial Ψ_{ave}*; seen in solvers*
- `weight_ttlen = 0.0`
  weight for coils length error, if zero, turned off; seen in length
- `target_length = 0.0`
  target value (or for normalization) of the coils length, if zero, automatically set to initial actual length; seen in rdcoils
- `weight_specw = 0.0`
  weight for spectral condensation error, if zero, turned off; seen in specwid; (not ready)
- `weight_ccsep = 0.0`
  weight for coil-coil separation constraint, if zero, turned off; seen in coilsep; (not ready)
- `weight_Inorm = 1.0`
  additional factor for normalizing currents; the larger, the more optimized for currents; seen in rdcoils
- `weight_Gnorm = 1.0`
  additional factor for normalizing geometric variables; the larger, the more optimized for coil shapes; seen in rdcoils

---

- `case_optimize = 1`
  specify optimizing options.
  -2: check the 2nd derivatives; seen infdcheck; (not ready)
  -1: check the 1st derivatives; seen infdcheck;
  0: no optimizations performed;

1: optimizing with algorithms using the gradient (DF and/or CG); seen in solvers;
2: optimizing with algorithms using the Hessian (HT and/or NT); seen in solvers; (not ready)

- `exit_tol = 1.000D-04`
  additional creteria to judge if the cost function decreases significantly; if $\frac{|\chi_i^2-\chi_{i-5}^2|}{\chi_i^2} < exit\_tol$*, send an exit signal; seen in solvers*
- `DF_maxiter = 0`
  maximum iterations allowed for using Differential Flow (DF); if zero, turned of; seen in descent
- `DF_xtol = 1.000D-08`
  relative error for ODE solver; seen in descent
- `DF_tausta = 0.000D+00`
  starting value of τ*; usually* 0.0 *is a good idea; seen in descent*
- `DF_tauend = 0.000D+00`
  ending value of τ*; the larger value of* $\tau_{end} - \tau_{sta}$*, the more optimized; seen in descent*
- `CG_maxiter = 0`
  maximum iterations allowed for using Conjugate Gradient (CG); if zero, turned of; seen in congrad
- `CG_xtol = 1.000D-08`
  the stopping criteria of finding minimum; if $|d\chi^2/d\mathbf{X}| < CG\_xtol$*, exit the optimization; seen in congrad;*
- `CG_wolfe_c1 = 1.000D-04`
  c1 value in the strong wolfe condition for line search; usually 1.0×10^{-4}*; seen in congrad;*
- `CG_wolfe_c2 = 0.1`
  c2 value in the strong wolfe condition for line search; if one CG step takes too long, try to increase c2, but remember $0 < c1 < c2 < 1$*; seen in congrad;*
- `LM_maxiter = 0`
  maximum iterations allowed for using Levenberg-Marquard (LM); if zero, turned of; seen in lmalg
- `LM_xtol = 1.000D-08`
  the stopping criteria of finding minimum; if the relative error between two consecutivec iterates is at most xtol, the optimization terminates; seen in lmalg;
- `LM_ftol = 1.000D-08`
  the stopping criteria of finding minimum; if both the actual and predicted relative reductions in the sum of squares are at most ftol, the optimization terminates; seen in lmalg;
- `LM_factor = 1.000D+02`
  *factor is a positive input variable used in determining the initial step bound. this bound is set to the product of factor and the euclidean norm of diag*x if nonzero, or else to factor itself. in most cases factor should lie in the interval (0.1,100.0). 100 is a generally recommended value. seen in lmalg;*

- `case_postproc = 1`
  specify post-processing options.
  0: no extra post-processing;
  1: evaluate the present coils for each cost functions, coil curvature, coil-coil separation, and coil-plasma separation, Bn harmonics overlap, coil importance;
  2: diagnos; write SPEC input file;
  3: diagnos; Field-line tracing, axis locating and iota calculating;
  4: diagnos; Field-line tracing; Calculates Bmn coefficients in Boozer coordinates;
- `update_plasma = 0`
  if euqals 1, write example.plasma file with updated Bn coefficients;
- `pp_phi = 0.0`
  Toroidal angle $\phi = pp\_phi * \pi$ *for filed-line tracing, axis locating, etc.*
- `pp_raxis = 0.0`
  `pp_zaxis = 0.0`
  Initial guess for axis positions (raxis, zaxis). If both zero, will be overide to $(\frac{r_1+r_2}{2}, \frac{z_1+z_2}{2})$*, where* $r_1 = R(0,\phi)$ *,* $r_2 = R(\pi,\phi)$ *(likewise for* z_1, z_2*.)*
- `pp_rmax = 0.0`
  `pp_zmax = 0.0`
  Upper bounds for field-line tracing. If both zero, will be overide to (r_1, z_1)*.*
- `pp_ns = 10`
  Number of surfaces for filed-line tracing, axis locating, etc. Starting points on ϕ *will be interpolated between* (r_{axis}, z_{axis}) *and* (r_{max}, z_{max})*.*

- `pp_maxiter = 1000`
  Cycles for tracing the field-line, representing the dots for each field-line in Poincare plots.
- `pp_tol = 1.0E-6`
  Tolerance of ODE solver used for tracing field-lines.
- `save_freq = 1`
  frequency for writing output files; should be positive; seen in solvers;
- `save_coils = 0`
  flag for indicating whether write example.focus *and* example.coils*; seen in saving*;
- `save_harmonics = 0`
  flag for indicating whether write example.harmonics*; seen in saving*;
- `save_filaments = 0`
  flag for indicating whether write .example.filaments.xxxxxx*; seen in saving*;

initial.h last modified on 019-04-23 10:15:20.; Focus subroutines;